Pattern Recognition

Song Bai

Email: songbai.site@gmail.com

What we are doing with Pattern Recognition

?

"Jack Ma"

Raw Data

Extract Informative Pattern Representation

PR Model (after training)

Recognition Results

Outlines

- Pattern Representation Learning
  - Unsupervised Representation Learning (PCA, NMF)
  - Supervised Representation Learning (LDA, GE)
  - Clustering and Applications
- Patter Recognition Models
  - Gaussian Mixture Model and Boosting
  - Support Vector Machines
  - Deep Learning (*a.k.a.* deep neural networks)

Textbooks and References

(no fixed textbook)

- Books
  - R. O. Duda, P. E. Hart & D.G. Stork,
    "**Pattern Classification**",
    John Wiley, 2001.
  - K. P. Murphy,
    “**Machine Learning: A Probabilistic Perspective**",
    MIT Press, 2012.
- References
  - Lists of important papers will be provided with some lectures
  - CVPR, ICML, etc

Outlines

- Representation Learning
  - Unsupervised Representation Learning (PCA, NMF)
  - Supervised Representation Learning (LDA, GE)
  - Clustering and Applications
- Patter Recognition Models
  - Gaussian Mixture Model and Boosting
  - Support Vector Machines
  - Deep Learning (*a.k.a.* deep neural networks)

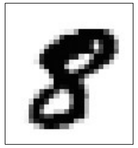
Unsupervised Feature Learning

Principal Component Analysis

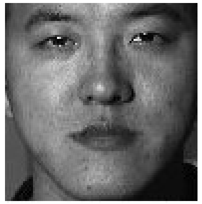
Representation Learning / Extraction

- Representation learning refers to transforming the raw data into another (possibly lower-dimensional) space.
- Such that, in the new space, one can perform pattern recognition more easily.
- Criterion for good representation in different problem settings:
  - **Unsupervised**: minimize information loss (no class information)
  - **Supervised**: maximize discrimination (with class information)

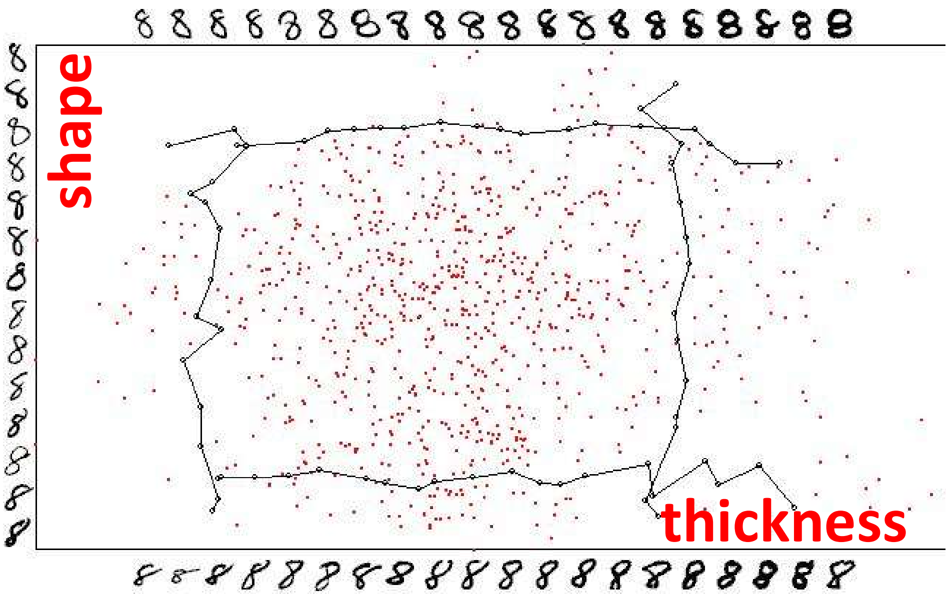
What is Feature Extraction



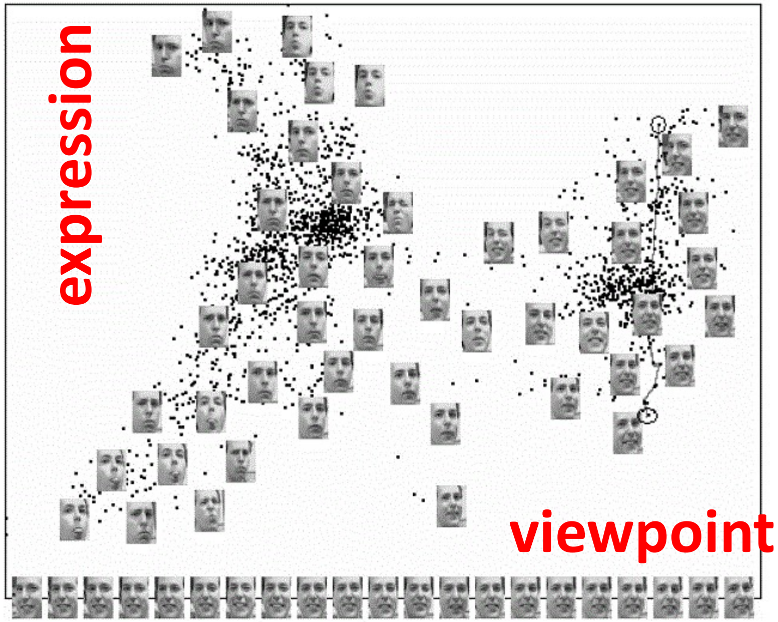
28 x 28 pixels



32 x 32 pixels



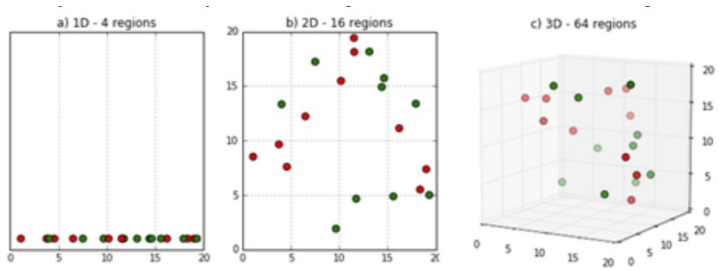
Features of Digit “8”



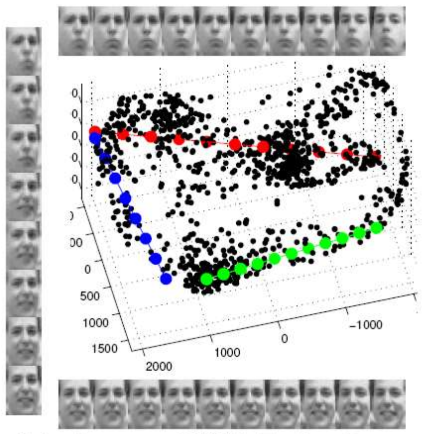
Features of Face

Why Feature Extraction?

- Many pattern recognition techniques may not be effective for high-dimensional data
  - Curse of dimensionality



- Patterns may have small intrinsic dimension
  - E.g., # genes responsible for a certain disease may be small
  - E.g., face images of one person captured with different illumination conditions

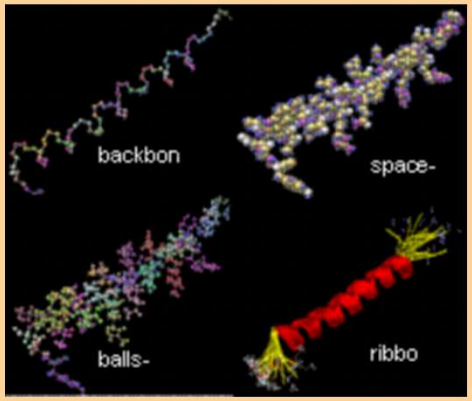


Why Feature Extraction?

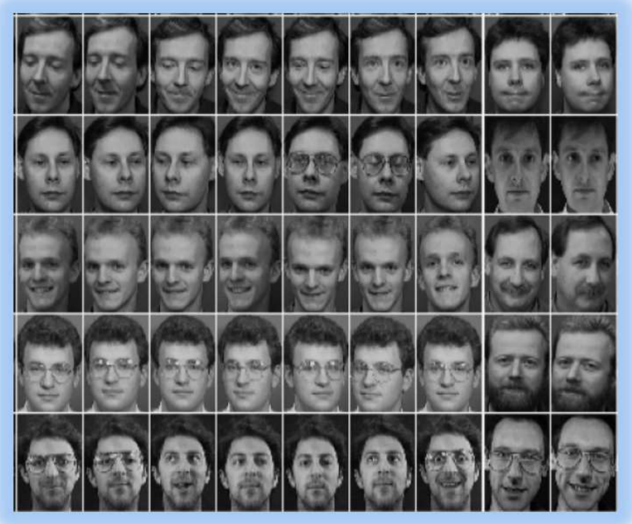
- Visualization: projecting high-dimensional data onto 2D or 3D planes
- Data compression: efficient storage and retrieval
- Noise removal: positive effect on testing accuracy

Applications of Feature Extraction

- Face recognition
- Handwritten digit recognition
- Text mining
- Image retrieval
- Protein classification



Proteins



Face Images

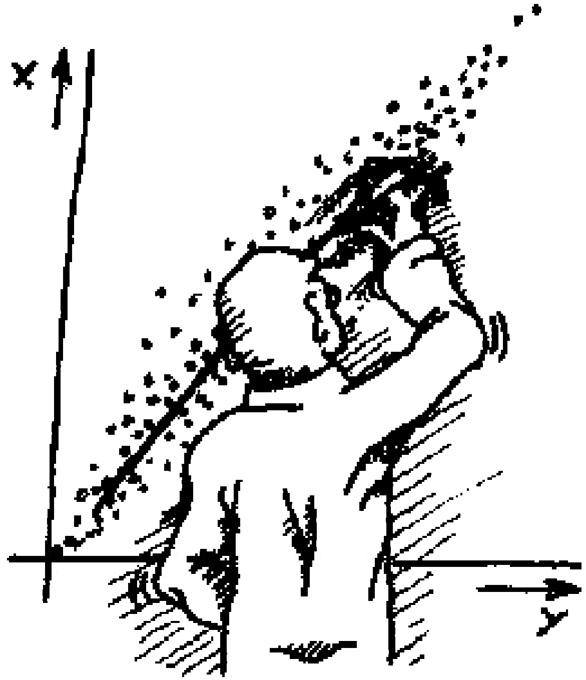
Feature Extraction Algorithms

- Unsupervised (this lecture)
  - Principal Component Analysis (PCA)
  - Nonnegative Matrix Factorization (NMF)
  - Independent Component Analysis (ICA) [Reading]

- Supervised (next lecture)
  - Linear Discriminant Analysis (LDA)
  - General Graph Embedding (GE)
  - Canonical Correlation Analysis (CCA) [Reading, encouraged]

Principal Component Analysis (PCA)

- Probably the most widely-used and well-known multivariate analysis method.
- Introduced by Pearson (1901)
- First applied in ecology by Goodall (1954) under the name “factor analysis”.
- *De facto* data pre-processing operation.

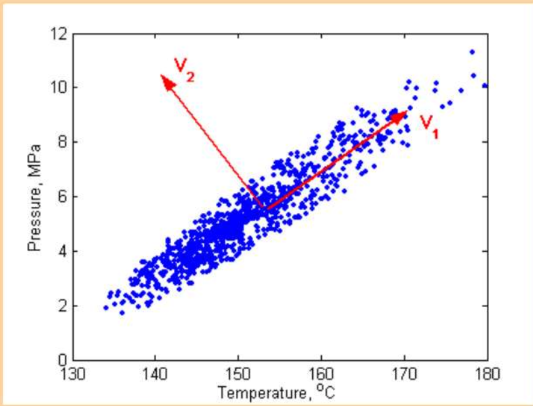


Least square error data fitting

Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points in Space" Philosophical Magazine **2** (11): 559–572.

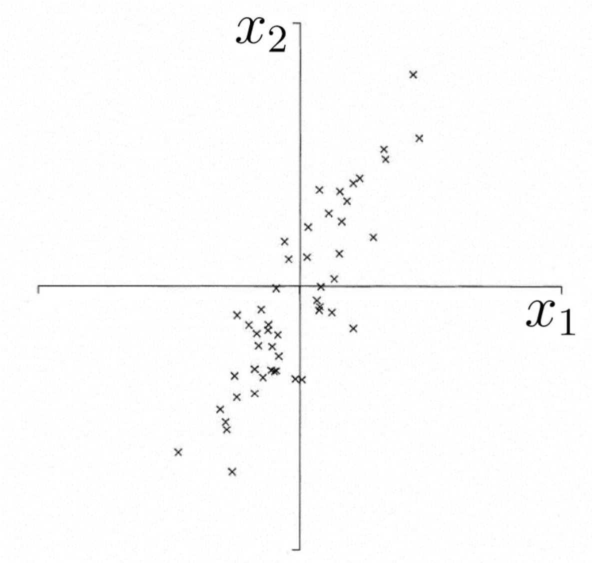
What is Principal Component Analysis?

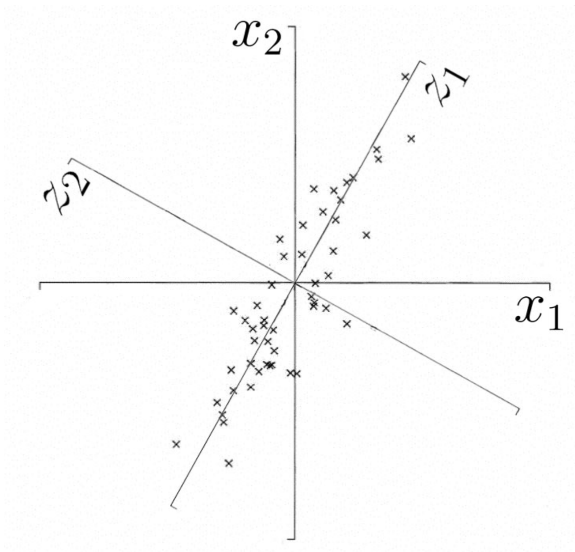
- Principal component analysis (PCA)
  - Reduce the dimensionality of a collection of observations by finding a new set of variables, smaller than the original set of variables
  - Capture big (principal) variability in the data and ignore small variability



- Variation in samples
  - The new variables, called principal components (PCs), are ordered by variations corresponding to different PCs.

Geometric Picture of Principal Components

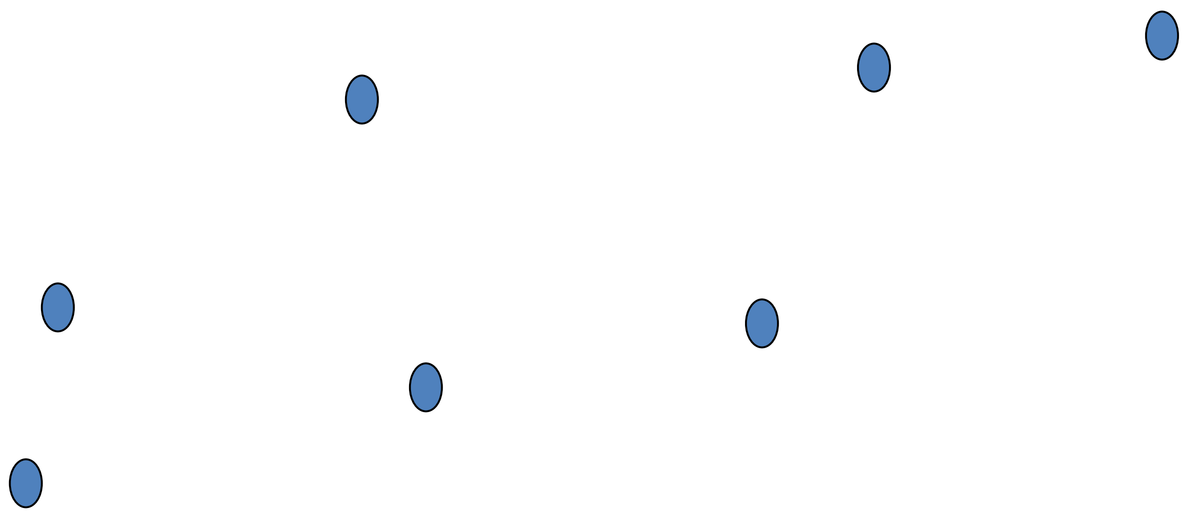




- The 1st PC z_1 is a minimum distance fit to a line in X space
- The 2nd PC z_2 is a minimum distance fit to a line in the plane orthogonal to the 1st PC

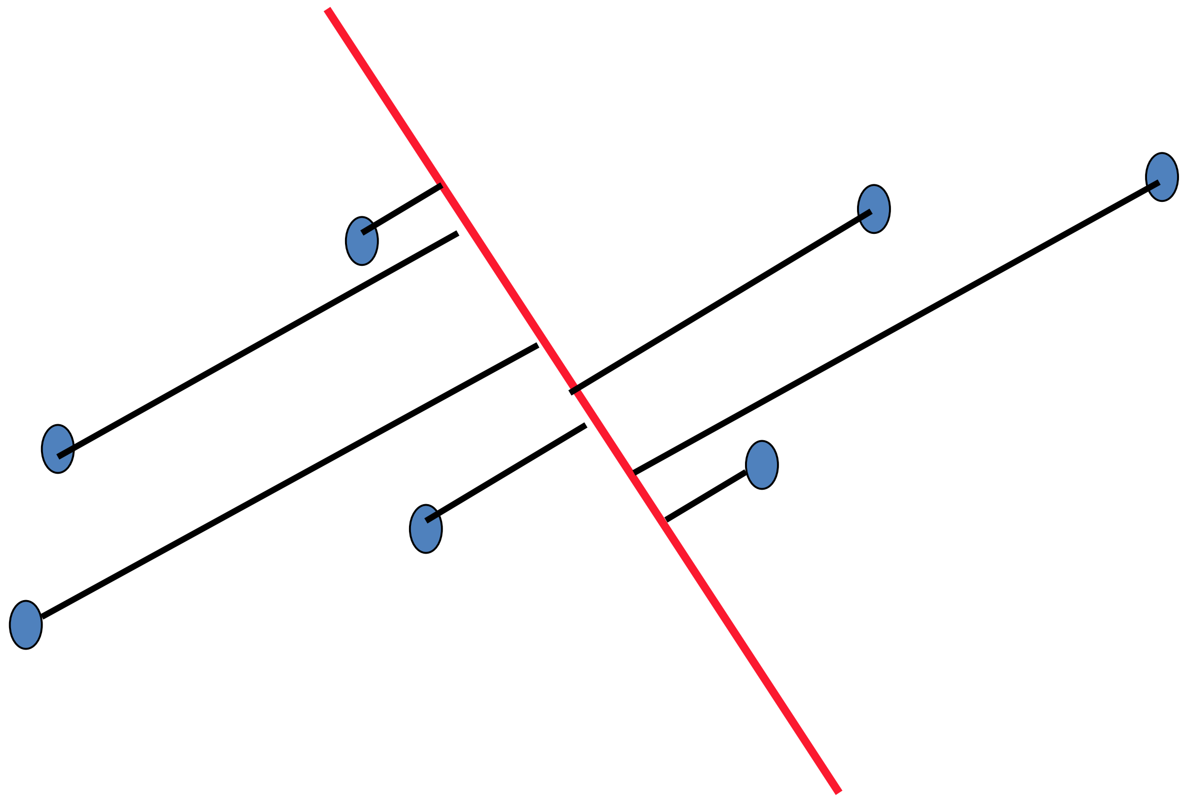
PCs are a series of linear least squares fits to a sample set, each orthogonal to all the previous ones.

Geometric Picture of Principal Components



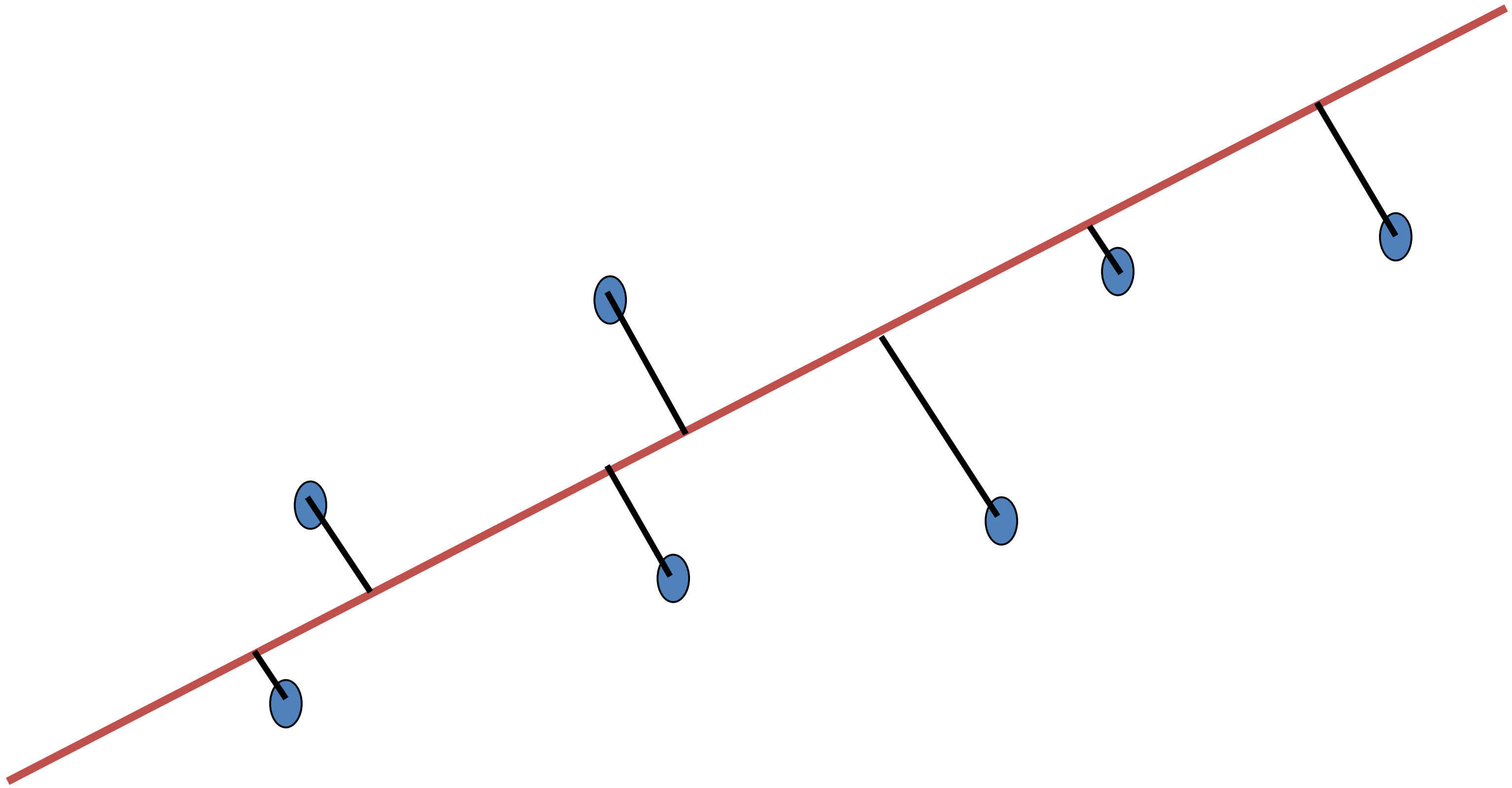
Sample Points

Geometric Picture of Principal Components



linear least squares fit: Large -> small variance

Geometric Picture of Principal Components



linear least squares fit: Small -> high variance

Algebraic Definition of PCs

Given a sample set of *n* observations on a vector of *d* variables

$$\{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d$$

define the first principal component by the linear projection a_1

$$z_1 = a_1^T x$$

where the vector $a_1 = (a_{11}, a_{21}, \ldots, a_{d1})^T$

is chosen such that $\mathrm{var}[z_1]$ is maximum.

Algebraic Definition of PCs

To find a_1 first note that

$$\mathrm{var}[z_1] = E((z_1 - \bar{z}_1)^2) = \frac{1}{n}\sum_{i=1}^{n}\left(a_1^T x_i - a_1^T \bar{x}\right)^2$$

$$= \frac{1}{n}\sum_{i=1}^{n} a_1^T \left(x_i - \bar{x}\right)\left(x_i - \bar{x}\right)^T a_1 = a_1^T S a_1$$

where $S = \frac{1}{n}\sum_{i=1}^{n} \left(x_i - \bar{x}\right)\left(x_i - \bar{x}\right)^T$

is the covariance matrix,

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the mean.

Algebraic Derivation of PCs

To find a_1 that maximizes $\text{var}[z_1]$ subject to $a_1^T a_1 = 1$

Let λ be a Lagrange multiplier

$$L = a_1^T S a_1 - \lambda(a_1^T a_1 - 1)$$
$$\Rightarrow \frac{\partial}{\partial a_1} L = S a_1 - \lambda a_1 = 0$$
$$\Rightarrow (S - \lambda I_d) a_1 = 0$$

therefore a_1 is an eigenvector of S

corresponding to the largest eigenvalue $\lambda = \lambda_1$.

Algebraic Derivation of PCs

Similarly, a_2 is also an eigenvector of S

whose eigenvalue $\lambda = \lambda_2$ is the second largest.

In general

$$\text{var}[z_k] = a_k^T S a_k = \lambda_k$$

- The k^{th} largest eigenvalue of S is the variance of the k^{th} PC.
- The k^{th} PC z_k retains the k^{th} greatest variation in the samples

Algebraic Derivation of PCs

- Main steps for computing PCs
  - Calculate the covariance matrix S.
  - Compute its eigenvectors: $\{a_i\}_{i=1}^{d}$
  - The first p eigenvectors $\{a_i\}_{i=1}^{p}$ form the p PCs.
  - The transformation matrix G consists of the p PCs:

$$G \leftarrow [a_1, a_2, \ldots, a_p]$$

$$y = G^T x$$

Practical Computation of PCA

- In practice, we compute the PCs via singular value decomposition (SVD) on the centered data matrix.

- Form the centered data matrix:

$$X_{d,n} = [(x_1 - \bar{x}), \ldots, (x_n - \bar{x})]$$

- Compute its SVD:

$$X = U_{d,d} D_{d,n} (V_{n,n})^T$$

- U and V are orthogonal matrices, D is a diagonal matrix

Practical Computation of PCA

- Note that the scatter/covariance matrix can be written as:

$$S = XX^T = UD^2U^T$$

- So the eigenvectors of S are the columns of U and the eigenvalues are the diagonal elements of D^2
  - why?

- Take only a few significant eigenvalue-eigenvector pairs p << d. The new reconstructed sample from low-dim space is:

$$x_i = \bar{x} + U_{d,p}U_{d,p}^T(x_i - \bar{x})$$

PCA and Classification

- Classification with PCA
  - Project both training and testing data into the PCs space
  - For each testing datum, use NN for classification
  - Issue: accuracy is sensitive to the number of PCs

How to determine the number of PCs?

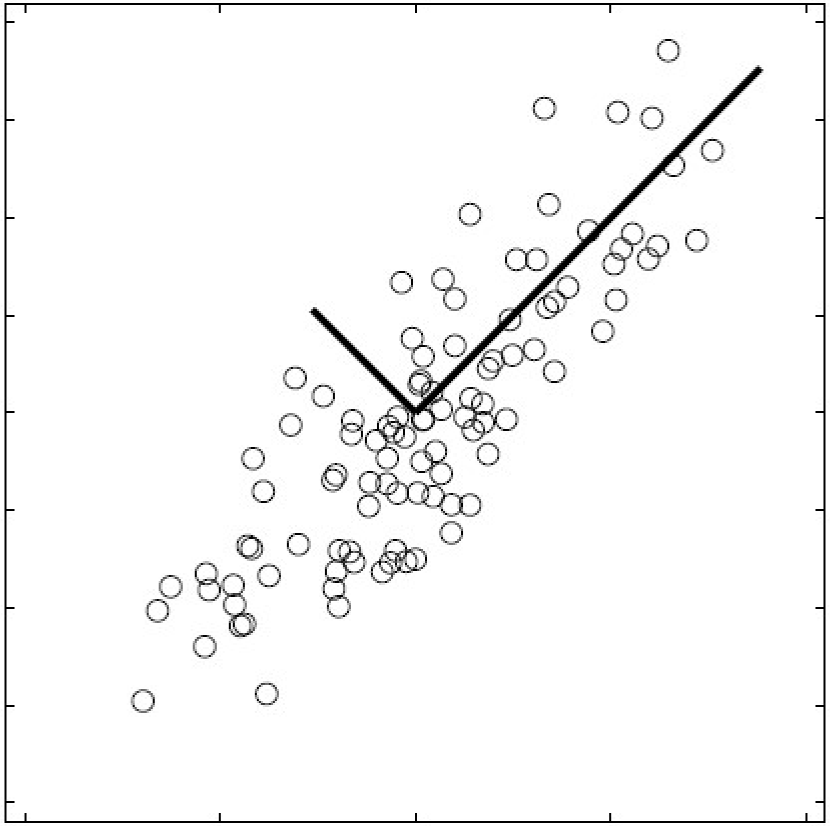
- PCA is not always an optimal feature extraction procedure for classification purpose
  - Suppose there are C classes in the training data
  - PCA is based on the sample covariance which characterizes the scatter of the entire data set, irrespective of class-membership
  - The projection axes chosen by PCA might not provide good discrimination power

How many principal components to keep?

- To choose *p* based on percentage of variation to retain, we can use the following criterion (smallest *p*):

$$\frac{\sum_{i=1}^{p} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq Threshold\ (e.g., 0.95)$$

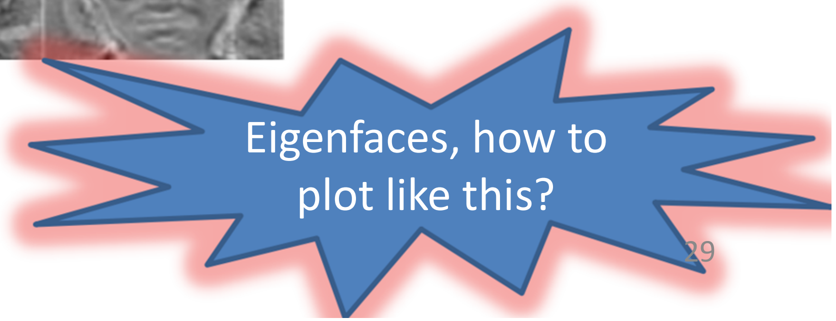
Visualize PCs



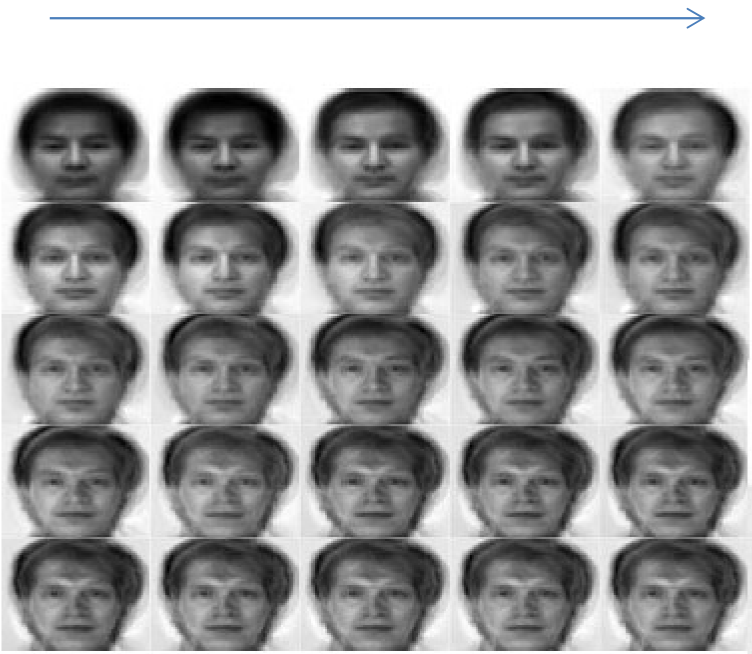
Data points are represented in a rotated orthogonal coordinate system: the origin is the mean of the data points and the axes are provided by the eigenvectors.

Visualize PCs

Face images



Reconstruction with PCs



Adding eigenfaces one at a time

Adding eigenfaces eight at a time

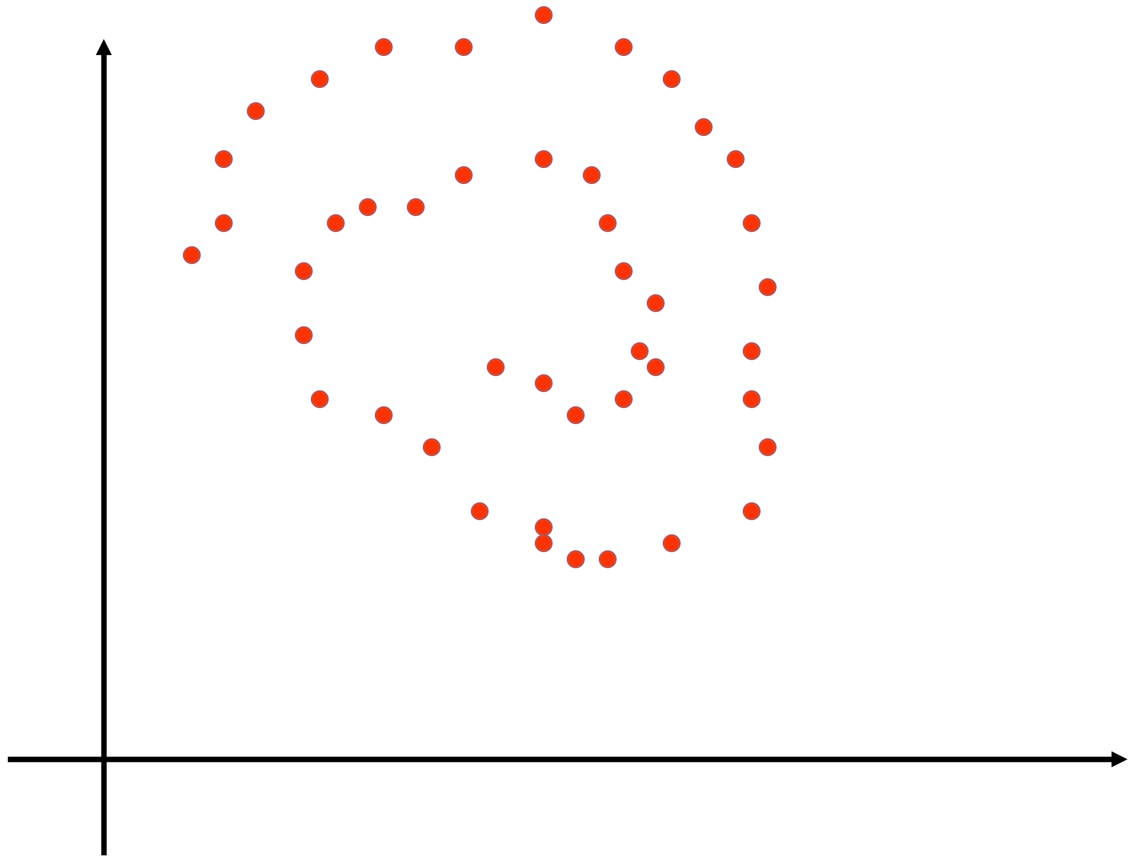
$$x_i = \bar{x} + U_{d,p} U_{d,p}^T (x_i - \bar{x})$$

PCA Remarks

- PCA
  - finds orthonormal basis for data
  - Sorts dimensions in order of “importance”
  - Discard low significance dimensions

- Uses:
  - Get compact description
  - Ignore noise
  - Improve classification (hopefully)

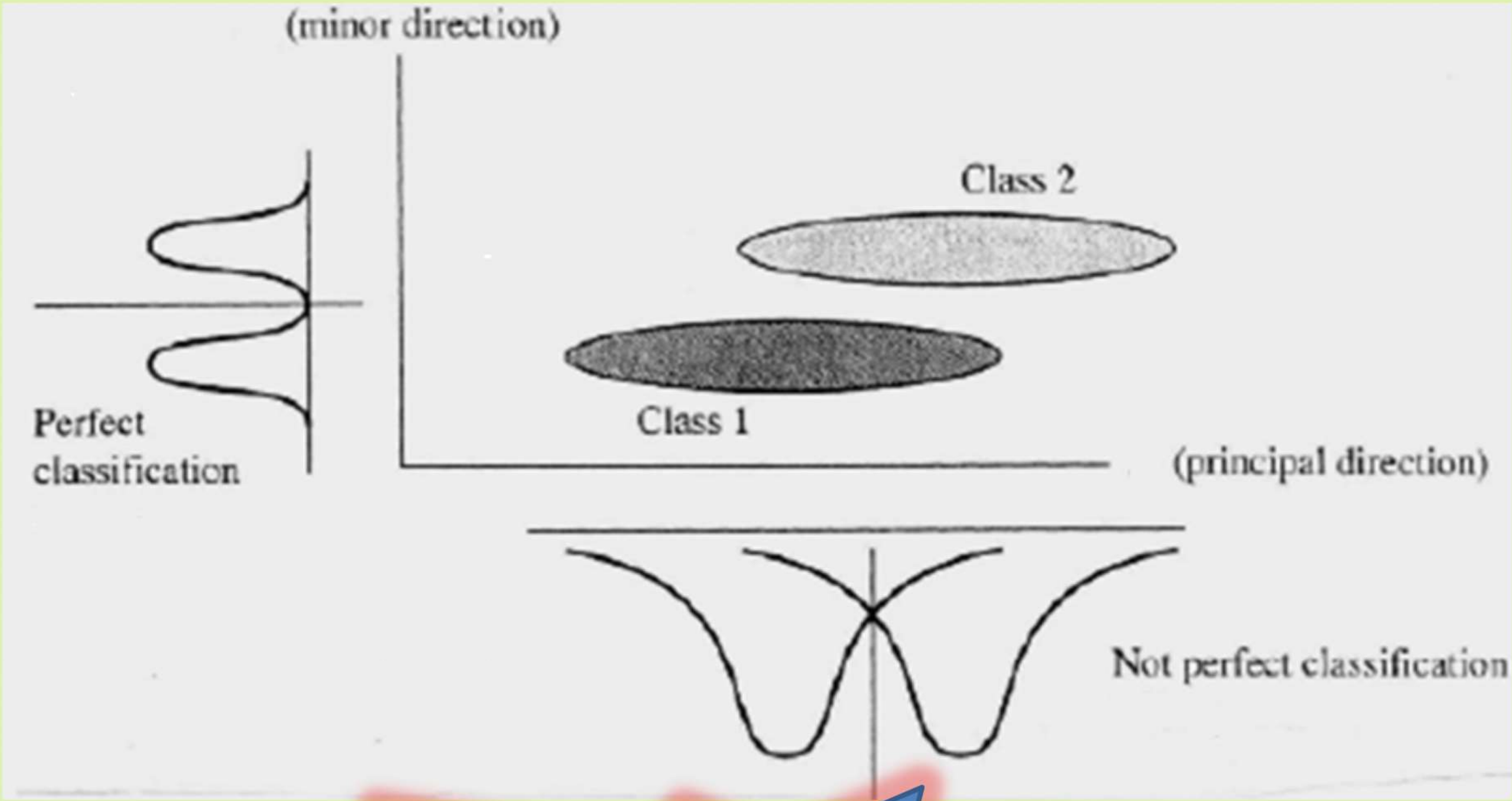
PCA Remarks



PCA cannot capture NON-LINEAR structure!

Note: Curvilinear Component Analysis can solve this case. Study this work if you are interested.

PCA doesn't know class labels



So LDA next week!

Summary of PCA

Algorithm 1 Algorithm for PCA

Input: Samples $\{x_1, x_2, \cdots, x_N\}$.

1. Compute the covariance matrix:

   $S = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T$;

2. Perform Eigenvalue Decomposition: $[U] = \mathrm{eig}(S)$;
3. Output PCs matrix $U(:, 1:p)$.

Discussions

- What can we do with PCA (given that it is generally worse for classification than other supervised algorithms)?

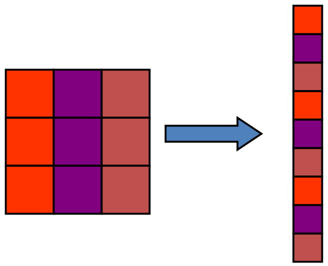
Unsupervised Feature Extraction II:
Nonnegative Matrix Factorization

A Quick Review of Linear Algebra

- Every vector can be expressed as the linear combination of basis vectors

$$\begin{bmatrix} 2 \\ 6 \end{bmatrix} = 2 \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 6 \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \text{ or } \begin{bmatrix} 2 \\ 6 \end{bmatrix} = 2 \cdot \begin{bmatrix} 1 \\ 3 \end{bmatrix} + 0 \cdot \begin{bmatrix} -6 \\ 2 \end{bmatrix}$$

- Can think of images as big vectors

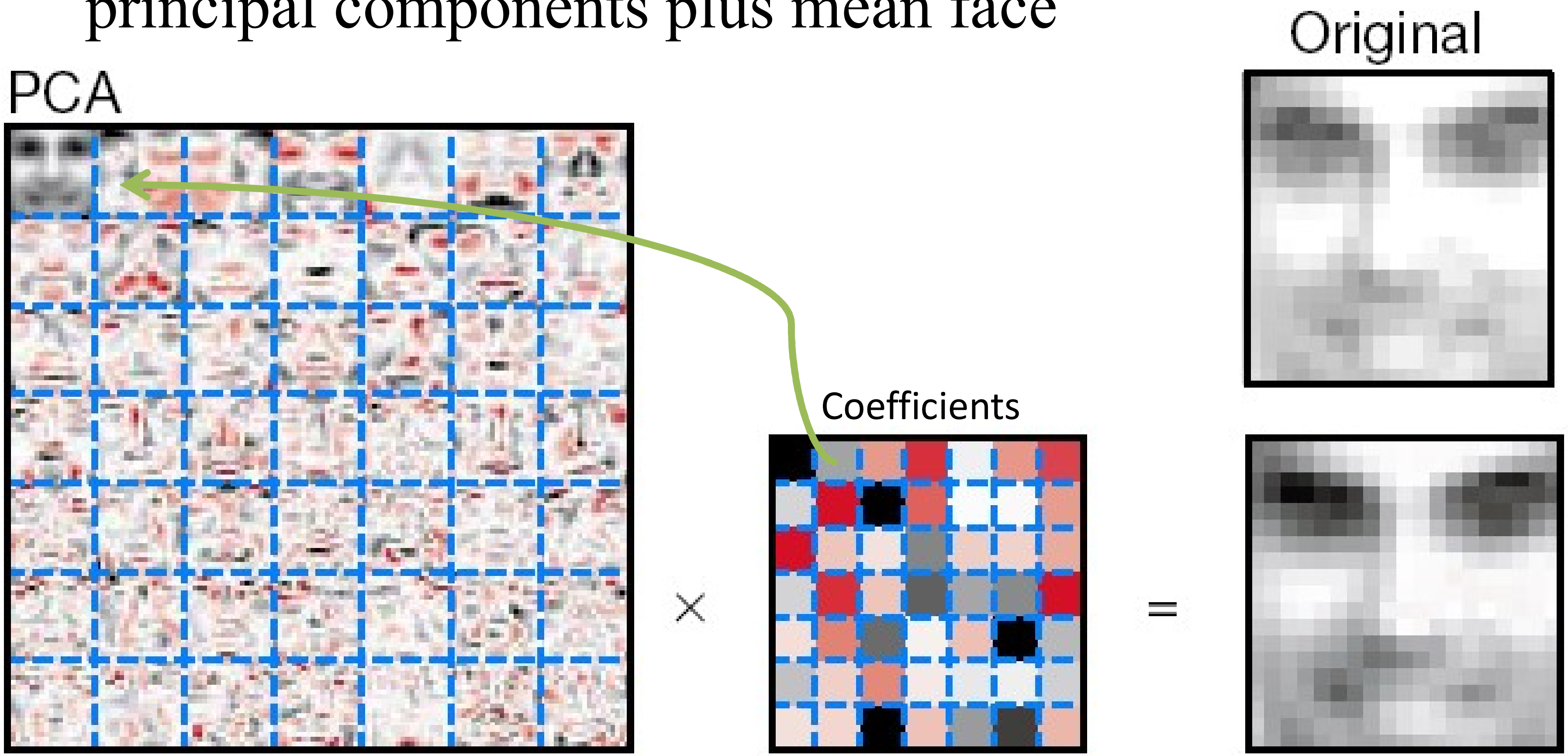


(Raster scan image into vector)

- This means we can express an image as the linear combination of a set of basis images

PCA Review

- Find a set of orthogonal principal components (basis)
- The reconstructed image is a linear combination of the principal components plus mean face

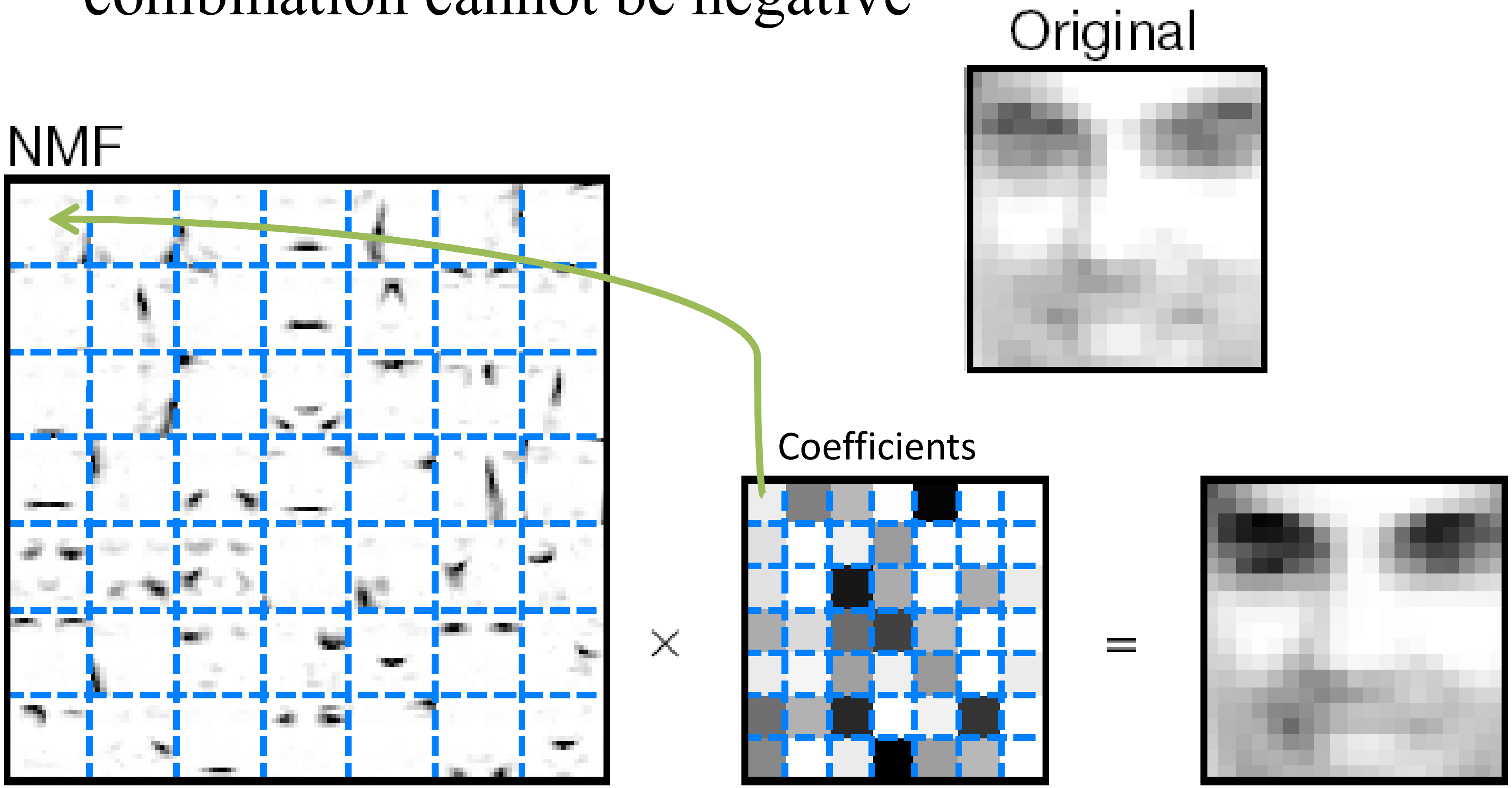


What we do not like about PCA?

- PCA involves adding up some basis vectors then subtracting others
- Basis vectors aren’t physically intuitive (negative) for many applications, e.g. documents
- Subtracting doesn’t make sense in context of some applications
  - How do you subtract a face?
  - What does subtraction mean in the context of document classification?

Non-negative Matrix Factorization

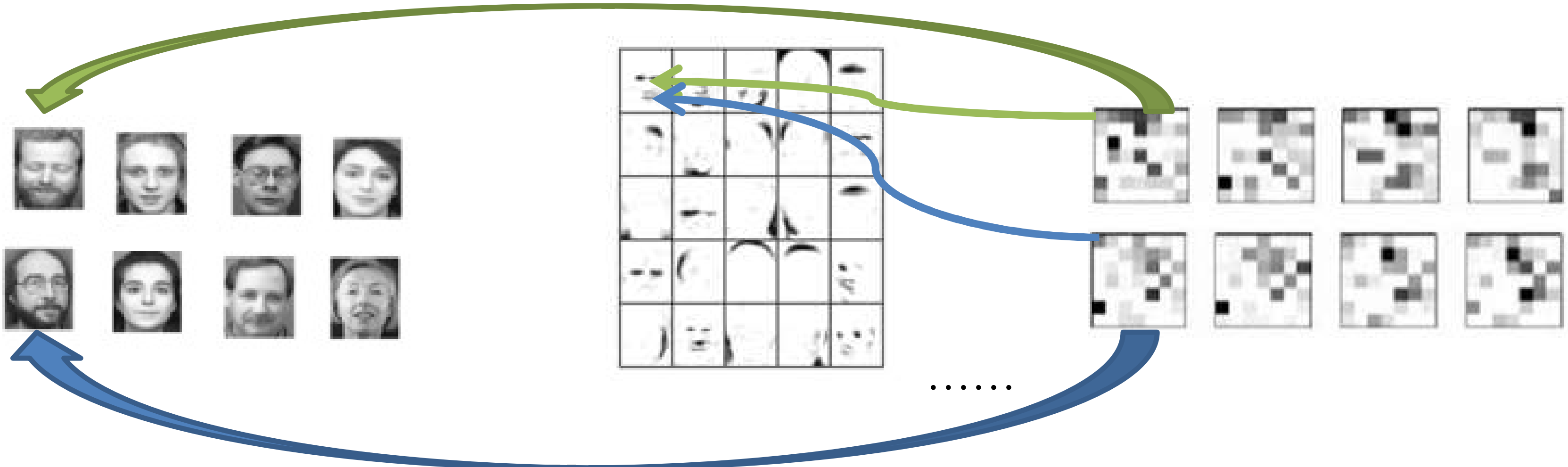
- Like PCA, except that the coefficients in the linear combination cannot be negative



Proposed by D. Lee and H. Seung (NIPS 2000)

Non-negative Matrix Factorization

- Matrix factorization: V≈WH
  - V: n×m matrix. Each column of which contains n nonnegative pixel values of one of the m facial images.
  - W: (n×r): r columns of W are called basis images.
  - H: (r×m): each column of H is called encoding.



V: an image is a column vector

W: a basis image is a column vector

H: a coefficient vector (shown as matrix here) is a column vector

NMF Basis Vectors

- Only allowing adding basis vectors makes intuitive sense
  - Has physical similarity in neurons
- Forcing the reconstruction coefficients to be nonnegative leads to **nice** basis vectors
  - To reconstruct vector (image), all you can do is to add in more basis vectors
  - This leads to basis vectors that represent parts

Objective Function

- Assume V is the sample matrix, the task is to approximate the original data matrix with two nonnegative data matrices:

$$\min_{W,H} \|V - WH\|^2 \quad s.t. \quad W \geq 0, H \geq 0.$$

- Let the value of a pixel in the original input image be $V_{i\mu}$. Let $(WH)_{i\mu}$ be the reconstructed pixel.

$$V_{i\mu} = (WH)_{i\mu} = \sum_{a=1}^{r} W_{ia} H_{a\mu}$$

How do we derive the update rules (H only, W similar)?

- Use gradient descent to find a local minimum

- The gradient descent update rule is:

$$H_{a\mu} \leftarrow H_{a\mu} + \eta_{a\mu} \left[(W^T V)_{a\mu} - (W^T W H)_{a\mu} \right]$$

Try by yourself

Deriving Update Rules (H only, W similar)

- Gradient Descent Rule:

$$H_{a\mu} \leftarrow H_{a\mu} + \eta_{a\mu} \left[(W^T V)_{a\mu} - (W^T W H)_{a\mu} \right]$$

- Set $\eta_{a\mu} = \dfrac{H_{a\mu}}{(W^T W H)_{a\mu}}$

- The update rule becomes

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^T V)_{a\mu}}{(W^T W H)_{a\mu}}$$

Try by yourself

What's significant about this?

- This is a multiplicative update
  - If the initial values of W and H are all non-negative, then the W and H can never become negative.

- This lets us produce a non-negative factorization

- See NIPS Paper for full proof that this will converge if you are interested.

http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.pdf

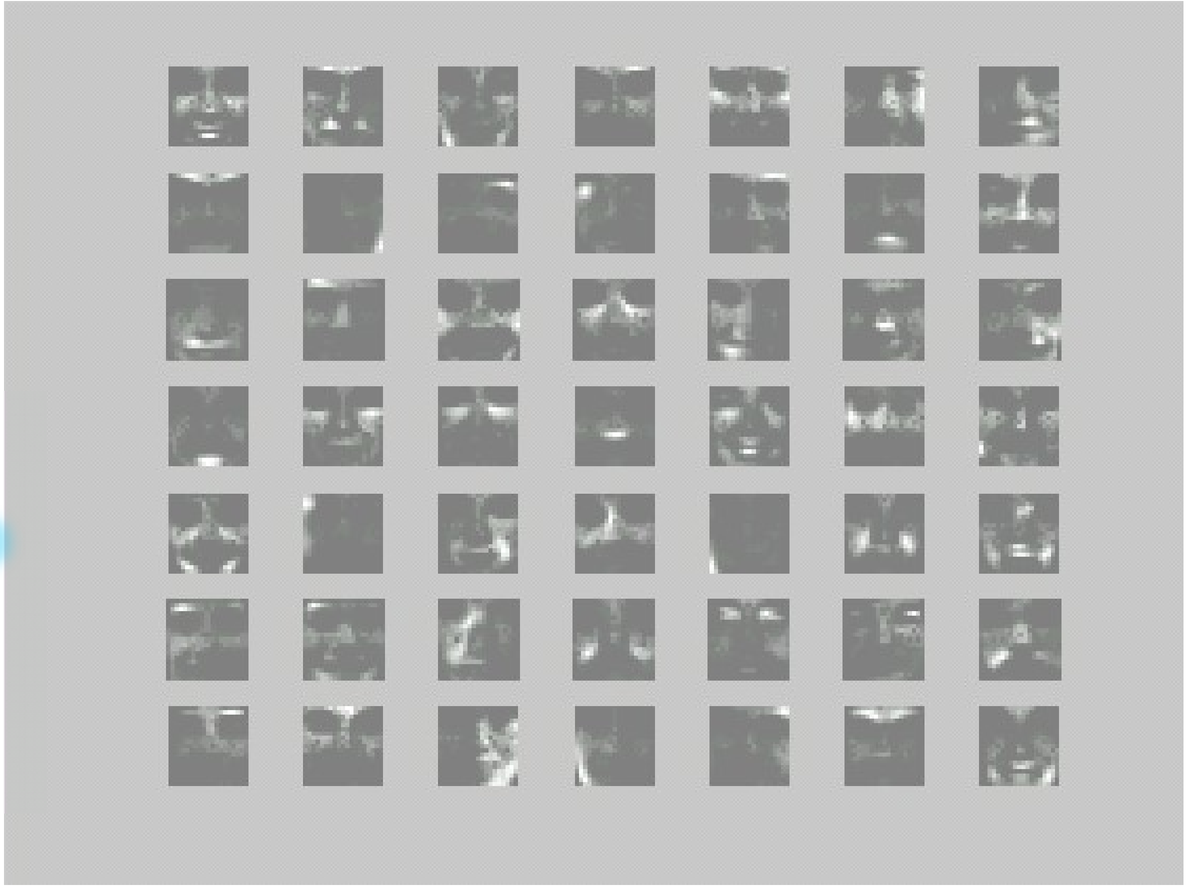
Example: Faces

- Training set: 2429 examples
- First 25 examples shown at right
- Set consists of 19x19 face images

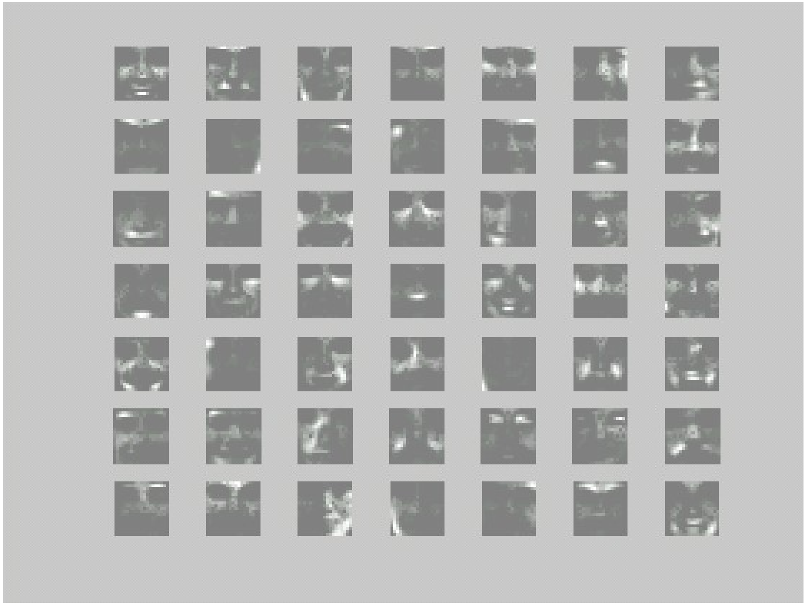
Example: Faces

- Basis Images:
  - Basis no.: 49
  - Iterations: 50

How to draw this figure?

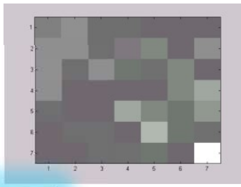


Face Reconstruction from Basis Vectors



W

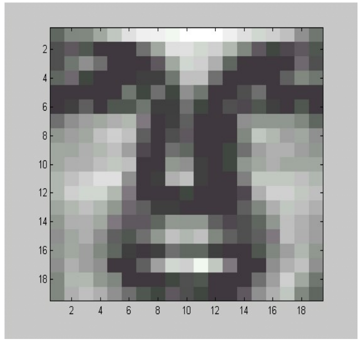
x

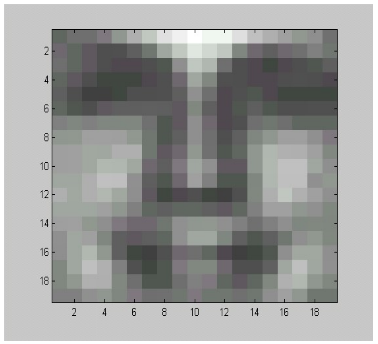


h

=

Original





W * h

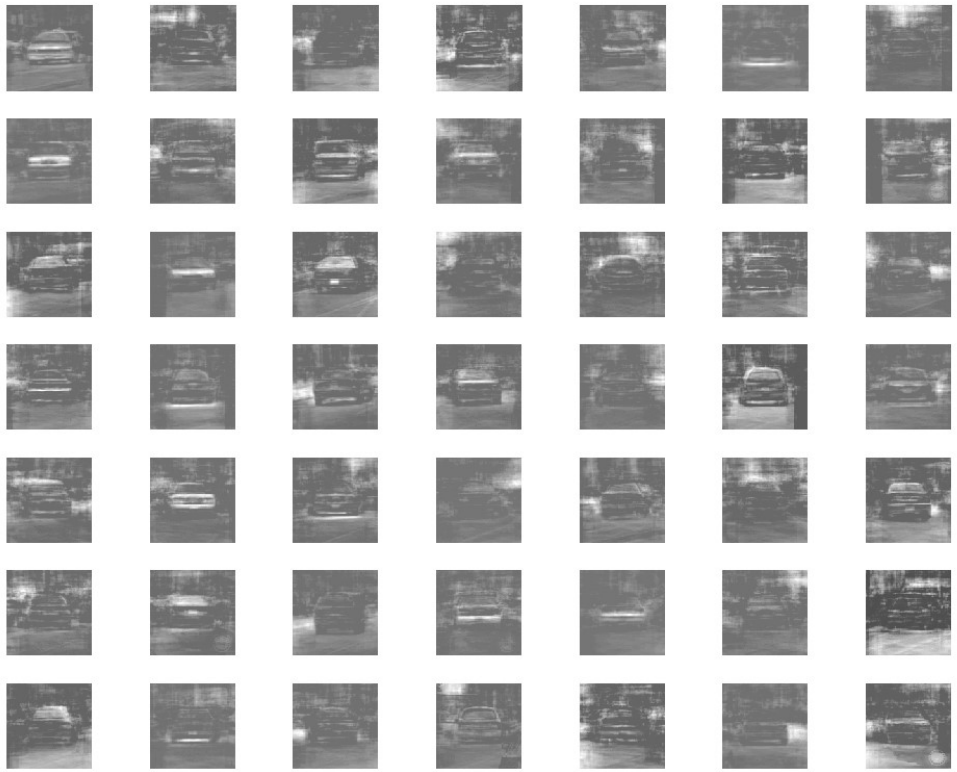
How to get h?

Example: Cars

- Training set: 200 examples
- First 25 examples shown at right
- Set consists of car images taken at various orientations

Example: Cars

- Basis Images
  - Basis no.: 49
  - Iterations: 310



Car Reconstruction from Basis Vectors

Originals (1-25)

Output (1-25)

Car Reconstruction from Basis Vectors

Original image

Reconstructed image

Why fence disappeared?

Discussions

- For new image, how to obtain the reconstruction coefficients?

- How to use NMF for classification, e.g. face recognition?

Summary of NMF

Algorithm 2 Algorithm for NMF

Input: Sample matrix $V = [v_1, v_2, \cdots, v_N]$.
Initialize W^0 and H^0 as arbitrary positive matrices.

for $t = 0 : 1 : T_{max}$ **do**

$H_{a\mu}^{t+1} = H_{a\mu}^{t} \frac{({W^t}^T V)_{a\mu}}{({W^t}^T W^t H^t)_{a\mu}}$;

$W_{a\mu}^{t+1} = W_{a\mu}^{t} \frac{(V {H^{t+1}}^T)_{a\mu}}{(W^t H^{t+1} {H^{t+1}}^T)_{a\mu}}$;

If $\|W^t - W^{t+1}\| < \epsilon$ and $\|H^t - H^{t+1}\| < \epsilon$

return;

end for

3. Output matrices W and H.

Discussions

- What are differences between NMF and PCA?

	NMF	**PCA**
Representation	Part-based	Holistic
Basis Image	Localized features	Eigenfaces
Constrains on W and H	Allow multiple basis images to represent a face but only additive combinations	Each face is approximated by a linear combination of all eigenfaces

Papers to Read and Self-Study

- D. Lee and H. Seung. [Algorithms for Non-negative Matrix Factorization](#) NIPS (2000).
- ICA (Independent Component Analysis): http://en.wikipedia.org/wiki/Independent_component_analysis
- CCA (Canonical Correlation Analysis): http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.6359&rep=rep1&type=pdf